The base station control unit is shown in Fig.5. The initial state of the base station control unit is the IDLE state. In normal operation mode the base station receives the Init_Command from RC and goes to the AUTH state (2). The AuthReq_Command is formed and sent to the RC.

The control unit stays at this state (3) until the AuthResp_Command is received or the timeout expires. If the timeout expires, the base station goes back to the IDLE state (4). When the AuthResp_Command is received, the authentication check is executed. If the authentication passes, the RC command is executed. After that, the Finish_Command is formed and sent to the RC. The control unit returns to the IDLE state (4).

If the alarm interrupt signal from the sensor's network is detected, the control unit moves to the ALARM state (5). In this state, the Alarm_command is periodically sent to the RC (the period is 1s). This state will be preserved (6) until the AlarmResp_Command from RC is received. After that, the control unit returns to the IDLE state (7).

Conclusion

This paper describes the communication module for an alarm system with a bidirectional interface. This module provides security on two levels. The first level is the interception-proof, noise-immune radio channel using the spread spectrum radio IC transceiver. The second level is the strong authentication for all RC commands. The authentication protocol is based on one-time passwords and a challenge-handshake procedure.

The authentication protocol implementation uses the RC5 encryption algorithm with a 40-bit key to satisfy US export restrictions. But key length can be easily increased up to 128 bits. Moreover, the whole algorithm can be easily replaced by any other customer-designed encryption algorithm with a 64-bit block size.

The functionality of the communication module can easily be modified and expanded. For example, it can be adapted to work with several RCs. To do this you would add a table with authorized RC serial numbers and corresponding secret keys to the firmware of the base station. The general system behavior can be changed so the system can be used in different keyless entry systems.

More detailed information about this project (including schematics and firmware) you can find in [5].

References

[1] Cypress CYWUSB6934 WirelessUSB™ Data Sheet (www.cypress.com)

[2] Cypress CYWUSB6953 WirelessUSB™ PRoC™ Data Sheet (www.cypress.com)

[3] Hardware Random Number Generator – AN2307 (www.cypress.com)

[4] Forward Error Correction using a WirelessUSB Radio System-on-Chip (SoC) Modem – AN2268, (www.cypress.com)

[5] Remote Keyless Entry Car Alarm with Floating Code – AN2308 (www.cypress.com)

Hardware Bitstream Sequence Recognizer

Oleksandr Karpin, Volodymyr Sokil

Computer Department, Lviv Polytechnic National University, S. Bandery Str., 12, Lviv, 79013, UKRAINE,
E-mail: karpinoo@yahoo.com, sokilvm@ukr.net

***Abstract* – this paper describes how to implement in hardware a bitstream sequence recognizer using the PSoC™ Pseudo Random Sequence Generator (PRS) User Module. The PRS can be used in digital communication systems with the serial data interface for automatic preamble detection and extraction, control words selection, etc.**

Key words – PSoC, digital communication system, microcontrollers.

I. Introduction

Most often communication systems use a serial interface for data transmission. In such cases, the bit flow must contain some auxiliary service bit sequences. For example, there might be a preamble sequence for receiver adjustment or a synchronization sequence (synchroword) for byte parsing.

The preamble sequences can be extracted both on the firmware or hardware levels, but receiver synchronization can only be performed in hardware. The common and necessary element of these applications is the bitstream sequence recognizer. An application of this type, with 8, 16, 24 or 32 bits, can easy be built on the PSoC™ device PRS User Module.

II. Recognizer implementation

The PRS block diagram is shown in Figure 1. It is a modular linear feedback shift register that generates a pseudo random bit sequence.

The Polynomial register value defines the internal block structure. If this register is initialized to a zero value, the PRS transfers to the Shift register (Figure 2). But this register has one very important feature – the value of this register can be compared with a predefined Seed register value. This is a wonderful base for the bits fragment recognizer.

The Shift register in the standard PRS mode has no data input and output connections. These connections must be defined manually by initialization of the special Control register. In this application, it is only necessary to route a register data input and clock to the internal structure of the target device. The clock source is easily set in PSoC Designer™ Device Editor. The input data source is defined by the PRSxx_x_INPUT_REG (DxBxxIN) register. All possible values of this register are shown in Table 1.

Note that the input data stream is inverted by the input exclusive-OR circuit. To return the input data to normal status, the Data Invert bit (0x80 mask) in the function register PRSxx_x_FUNC_REG (DxBxxFN) must be set.

The CompareType parameter can be set to “Equal,” “Less Then” or “Less Then or Equal” in the Device Editor. In our case, set the CompareType parameter to “Equal.”

The bitstream sequence, which needs to be recognized, is loaded into the PRS Seed register. If the input bitstream contains the defined bits, the Seed register and Shift register values will be equal to each other. The CompareOut parameter signal goes to active high.

But note that upon startup, the Seed register value is copied into the Shift register, causing the compare circuit to immediately trigger. So the first CompareOut signal after the module has started must be skipped.

TABLE 1

INPUT DATA SOURCE CONNECTION

Value	Connection
0x00	Low level (0)
0x10	High level (1)
0x20	Row Broadcast Net
0x30	Chain Function To Previous Block
0x40	Analog Column Comparator 0
0x50	Analog Column Comparator 1
0x60	Analog Column Comparator 2
0x70	Analog Column Comparator 3
0x80	Row Output 0
0x90	Row Output 1
0xA0	Row Output 2
0xB0	Row Output 3
0xC0	Row Input 0
0xD0	Row Input 1
0xE0	Row Input 2
0xF0	Row Input 3

The CompareOut parameter can be used as a control signal for different hardware receivers (for example, as a slave select signal for the SPIS User Module). Moreover, the interrupts can be used to control firmware. To enable the interrupts, the corresponding block mask must be set in the Interrupt Mask Register, INT_MSKx. The interrupt is triggered by the rising edge of the CompareOut signal.

III. Practical example

Let’s briefly consider a practical example of the bitstream sequence recognizer application – a serial bit receiver with automatic preamble synchronization. The test communication system uses two-wire serial interface with clock and data signals. The transmitter inserts 16 bits of preamble in the output bitstream. The first byte (0xFF) is the synchronization flag, the second byte (0x0F) is the “dummy” for receiver resynchronization. This byte is omitted by the receiver.

The receiver’s flowchart is shown in Figure 3.

The SPIS User Module is used as the hardware receiver. For byte synchronization, the preamble separator

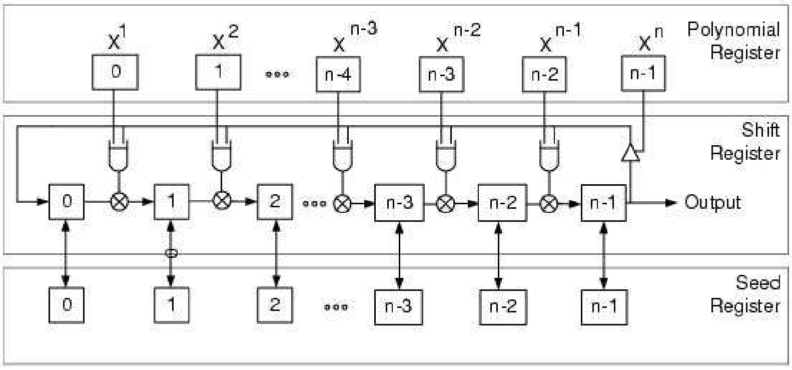

Fig. 1 PRS User Module Block Diagram, Data Width n = 8, 16, 24, 32

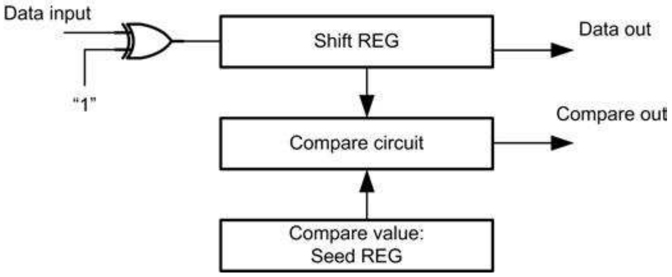

Fig. 2 PRS User Module Block Diagram, Polynomial Register = 0x00, Data Width n = 8

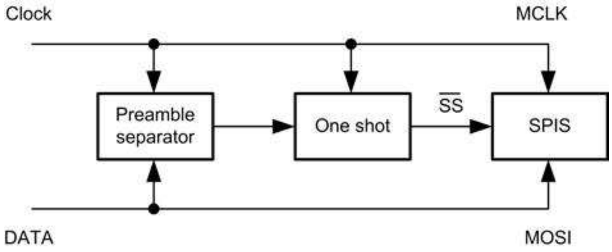

Fig. 3 Receiver Flowchart

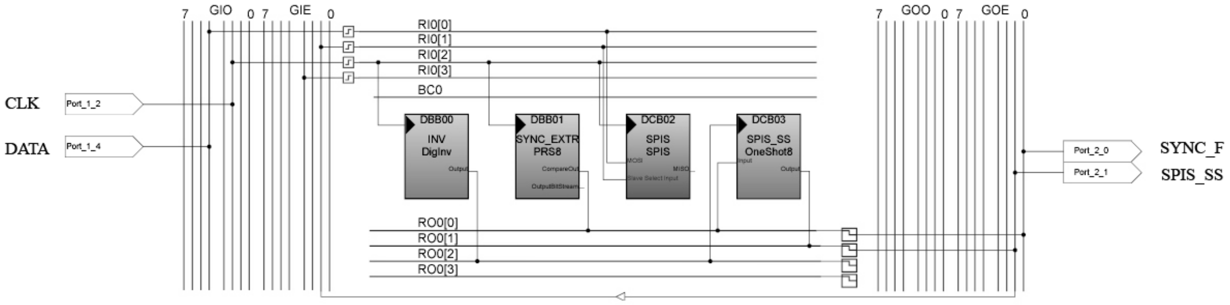

Fig. 4 PSoC Receiver Internal User Module Configuration

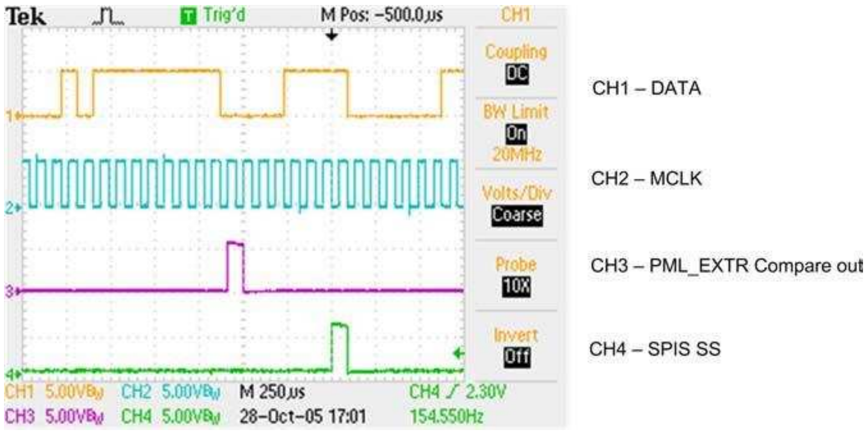
Tek
Trig'd
M Pos: -500.0µs
CH1
Coupling
DC
BW Limit
On
20MHz
Volts/Div
Coarse
Probe
10X
Invert
Off
CH1 5.00VBW CH2 5.00VBW M 250µs CH4 ⟋ 2.30V
CH3 5.00VBW CH4 5.00VBW 28-Oct-05 17:01 154.550Hz
CH1 – DATA
CH2 – MCLK
CH3 – PML_EXTR Compare out
CH4 – SPIS SS

Fig. 5 Receiver Preamble Synchronization Waveform

CompareOut signal must be delayed for 6 ½ MCLK periods. This delay is provided by the one-shot user module, which uses the inverted clock signal. Implementation and functioning principles of the one-shot user module are described in [1]. The delayed signal is used as the SPIS slave select input signal.

The internal structure of the receiver and oscilloscope waveforms for such implementation are shown in Figures 4 and 5, accordingly.

Conclusion

In this paper implementation in hardware a bitstream sequence recognizer using the PSoC™ Pseudo Random Sequence Generator (PRS) User Module is described.

References

[1] I. Mamontov, "PRS User Module as a One-Shot Pulse Width Discriminator and Debouncer" *AN2249, www.cypress.com*, June, 2005.